What is claimed is:

1. A method for reducing the restart time for a parallel application, the parallel application including a plurality of parallel operators, the method comprising

repeating the following:

setting a time interval to a next checkpoint;

waiting until the time interval expires;

sending checkpoint requests to each of the plurality of parallel operators; and

receiving and processing messages from one or more of the plurality of parallel operators.

2. The method of claim 1 further comprising

before entering the repeat loop:

receiving a ready message from each of the plurality of parallel operators indicating the parallel operator that originated the message is ready to accept checkpoint requests.

3. The method of claim 1 wherein receiving and processing messages from one or more of the plurality of parallel operators comprises

receiving a checkpoint information message, including checkpoint information, from one of the plurality of parallel operators; and

storing the checkpoint information, along with an identifier for the one of the parallel operators, in a checkpoint data store.

4. The method of claim 1 wherein receiving and processing messages from one or more of the plurality of parallel operators comprises

receiving a ready to proceed message from one of the plurality of parallel operators;

marking the one of the plurality of parallel operators as ready to proceed; and

if all of the plurality of parallel operators has been marked as ready to proceed, marking a current checkpoint as good.

5. The method of claim 1 wherein receiving and processing messages from one or more of the plurality of parallel operators comprises

receiving a checkpoint reject message from one of the plurality of parallel operators;

sending abandon checkpointing messages to the plurality of parallel operators; and

scheduling a new checkpoint.

6. The method of claim 1 wherein receiving and processing messages from one or more of the plurality of parallel operators comprises

receiving a recoverable error message from one or more of the plurality of parallel operators;

sending abandon checkpointing messages to the plurality of parallel operators;

waiting for ready messages from all of the plurality of parallel operators; and

scheduling a new checkpoint.

7. The method of claim 1 where receiving and processing messages from one or more of the plurality of parallel operators comprises

receiving a non-recoverable error message from one of the plurality of parallel operators; and

sending terminate messages to the plurality of parallel operators.

8. The method of claim 7 further comprising restarting the plurality of parallel operators.

9. The method of claim 8, where restarting comprises

sending initiate restart messages to the plurality of parallel processors; and

processing restart messages from the plurality of parallel processors.

10. The method of claim 9 where processing restart messages comprises

receiving an information request message from one or more of the plurality of parallel operators;

retrieving checkpoint information regarding the one or more of the plurality of parallel operators from the checkpoint data store; and

sending the retrieved information to the one of the plurality of parallel operators.

11. The method of claim 9 where processing restart messages comprises

receiving a ready to proceed message from one of the plurality of parallel operators;

marking the one of the plurality of parallel operators as ready to proceed; and

sending proceed messages to all of the plurality of parallel operators if all of the plurality of parallel operators have been marked as ready to proceed.

12. The method of claim 9 where processing restart messages comprises

receiving an error message from one or more of the plurality of parallel operators; and

terminating the processing of the plurality of parallel operators.

13. A method for one of a plurality of parallel operators to record its state, the method comprising

receiving a checkpoint request message on a control data stream;

waiting to enter a state suitable for checkpointing; and

sending a response message on the control data stream.

14. The method of claim 13 wherein waiting to enter a state suitable for checkpointing comprises

receiving a checkpoint marker on an input data stream;

finishing writing data to an output data stream; and

sending a checkpoint marker on the output data stream.

15. The method of claim 13 wherein waiting to enter a state suitable for checkpointing comprises

waiting for all of the parallel operator's outstanding input/output requests to be processed.

16. The method of claim 13 further comprising

determining that the parallel operator is not in a state suitable for checkpointing; and wherein sending a response message on the control data stream comprises

sending a checkpoint reject message on the control data stream.

17. The method of claim 16 further comprising

experiencing a recoverable error; and wherein sending a response message on the control data stream comprises

sending a recoverable error message on the control data stream.

18. The method of claim 16 further comprising

experiencing a non-recoverable error; and wherein sending a response message on the control data stream comprises

sending a non-recoverable error message on the control data stream.

19. A computer program, stored on a tangible storage medium, for use in reducing the restart time for a parallel application, the parallel application comprising a plurality of parallel operators, the computer program comprising

a CRCF component which includes executable instructions that cause a computer to repeat the following:

set a time interval to a next checkpoint;

wait until the time interval expires;

send checkpoint requests to the plurality of parallel operators;

receive and process messages from one or more of the plurality of parallel operators;

a plurality of parallel components, each of which is associated with one of the plurality of parallel operators, and each of which includes executable instructions that cause a computer to:

receive a checkpoint request message from the CRCF;

wait to enter a state suitable for checkpointing; and

send a checkpoint response message to the CRCF.

20. The computer program of claim 19 wherein

each of the parallel components include executable instructions that cause a computer to:

determine that the parallel operator is not in a state suitable for checkpointing; and, in sending a response message to the CRCF, the parallel component associated with that parallel operator causes the computer to

send a checkpoint reject message to the CRCF;

in receiving and processing messages from one or more of the plurality of parallel operators, the CRCF causes the computer to:

receive the checkpoint reject message; and

send abandon checkpoint messages to the plurality of parallel operators in response to the checkpoint reject message.

21. The computer program of claim 19 wherein

each of the parallel components include executable instructions that cause a computer to:

determine that one or more of the parallel operators has experienced a recoverable error; and, in sending a response message to the CRCF, the parallel component or

components associated with the one or more parallel operators that experienced the recoverable error or errors cause the computer to:

send a recoverable error message to the CRCF;

proceed with recovery; and

send a ready message to the CRCF;

in receiving and processing messages from one or more of the plurality of parallel operators, the CRCF causes the computer to:

receive the recoverable error message;

send abandon checkpoint messages to the plurality of parallel operators in response to the recoverable error message;

wait for the ready messages;

receive the ready messages; and

schedule a checkpoint.

22. The computer program of claim 19 wherein

each of the parallel components include executable instructions that cause a computer to:

determine that one of the parallel operators has experienced a non-recoverable error; and, in sending a response message to the CRCF, the parallel component associated with the one parallel operator causes the computer to:

send a non-recoverable error message to the CRCF;

in receiving and processing messages from one or more of the plurality of parallel operators, the CRCF causes the computer to:

receive the non-recoverable error message; and

send stop processing messages to the plurality of parallel operators in response to the non-recoverable error message.

23. The computer program of claim 22 wherein the CRCF further includes executable instructions that cause the computer to:

send an initiate restart message to one of the plurality of parallel operators.

24. The computer program of claim 23 wherein

in response to the restart message from the CRCF, the parallel component associated with the one parallel operator causes the computer to:

send an information request to the CRCF;

in responding to the information request, the CRCF causes the computer to:

retrieve checkpoint information regarding the one parallel operator from a checkpoint data store; and

send the checkpoint information to the one parallel operator.

25. The computer program of claim 23 wherein

the parallel component associated with one of the parallel operators further comprises executable instructions that cause the computer to:

send a ready to proceed message to the CRCF;

in responding to the ready to proceed message, the CRCF causes the computer to:

mark the one parallel operator as ready to proceed; and

if all of the plurality of parallel operators have been marked as ready to proceed, sending proceed messages to all of the plurality of parallel operators.

26. The computer program of claim 23 wherein

the parallel component associated with one of the parallel operators further comprises executable instructions that cause the computer to:

send an error message to the CRCF;

in responding to the error message, the CRCF causes the computer to:

send messages to all of the parallel operators to terminate their processing.